OPENMP COMPUTING OF A REFERENCE SOLUTION FOR COUPLED LORENZ SYSTEM ON [0,400]

I. G. Hristov[1,a], R. D. Hristova[1], S. N. Dimova[1], P. R. Armyanov[1], N. G. Shegunov[1], I. V. Puzynin[2], T. P. Puzynina[2], Z. A. Sharipov[2], Z. K. Tukhliev[2]

[1] *Sofia University "St. Kliment Ohridski", FMI, Sofia, Bulgaria*

[2] *Joint Institute for Nuclear Research, MLIT, Dubna, Russia*

E-mails: [a] ivanh@fmi.uni-sofia.bg

Obtaining a long term reference trajectory on the chaotic attractor for the coupled Lorenz system is a difficult task, due to the sensitive dependence on the initial conditions. Using the standard double-precision floating point arithmetic, we cannot obtain a reference solution longer than 2.5 time units. Combining OpenMP parallel technology together with GMP library (GNU multiple precision library), we parallelize the Taylor series algorithm for the coupled Lorenz system and obtain a reference solution in the rather long time interval - [0,400]. We performed two large computations, each using one CPU-node (32 cores), based on two Intel® Haswell processors. First computation was with 2158 decimal digits of precision and 2480-th order method, and second computation was for verification - with 2254 decimal digits of precision and 2580-th order method. The needed time for the second (larger) computation was 6.3 days. The parallel speedup when using these 32 cores is 23.1 with parallel efficiency 72.1 %.

Keywords: Variable step-size Taylor series method, Clean numerical simulation, Multiple precision, OpenMP parallel technology, Coupled Lorenz system

Ivan Hristov, Radoslava Hristova, Stefka Dimova, Petar Armyanov, Nikolay Shegunov, Igor Puzynin, Taisiya Puzynina, Zarif Sharipov, Zafar Tukhliev

1. Introduction

To compute a reliable long term trajectory for a chaotic dynamic system, we need a multiple precision floating point arithmetic library in order to deal with the sensitive dependence on the initial conditions. This is not enough, we also need a numerical method that steps efficiently at the level of the high precision, i.e. we need a method that allows arbitrary high order of accuracy. In this paper we use a numerical procedure called "Clean numerical simulation" to obtain verified long term trajectories for chaotic dynamical systems [1]. The procedure is based on the multiple precision Taylor series method [2]. In the case of very long time of integration, the computational problem can become pretty large and we need a parallelization of the algorithm. Although in our recent work [3] we reported a general MPI+OpenMP parallelization for the classical Lorenz system, pure OpenMP parallelization has its own importance and deserves special attention. In this paper we combine OpenMP parallel technology together with GMP library [4] to parallelize the Taylor series method and compute a reliable trajectory on the chaotic attractor for the more difficult coupled Lorenz system. The work can be regarded as a continuation and improvement of the results in our previous work [5].

2. The model problem – coupled Lorenz system

We consider as a model problem the coupled Lorenz system from [6]:

$$\begin{aligned} &\frac{dx}{dt} = a\,(y - x), && \frac{dy}{dt} = r_s x - y - xz - \varepsilon_s XY, && \frac{dz}{dt} = xy - bz \\ &\frac{dX}{dt} = ca\,(Y - X), && \frac{dY}{dt} = c\big(r_f X - Y - XZ\big) + \varepsilon_f Xy, && \frac{dZ}{dt} = XY - bZ \end{aligned} \qquad (1)$$

where a = 10, b = 8/3, c = 10, r_s = 28, r_f = 45, ε_s = 10^{-2}, ε_f = 10. The first three and last three equations are called slow and fast dynamics respectively. The first three equations (without the last term in the second equation) exactly represent the classical Lorenz system. For the above parameters the system has a chaotic attractor, which means a sensitive dependence on the initial conditions. This dependence is described by the relation $\delta(t) \sim \delta(0)e^{\lambda t}$, where λ is the maximum Lyapunov exponent and $\delta(t)$ is the distance between two adjacent trajectories. Solving the above relation with respect to t, we obtain the following relation for the predictability horizon (the Lyapunov time *T*): $T\sim \frac{1}{\lambda}\, ln\left(\frac{tol}{\varepsilon}\right)$. Here *tol* is our tolerance and ε is the round-off unit (precision). For the coupled Lorenz system λ = 11.5. If we use the standard double precision ($\varepsilon = 2^{-53}$) and tolerance $tol = 10^{-3}$, then $T \sim 2.5$. This means that with the general double-precision arithmetic we cannot obtain a reliable solution longer than 2.5 time units. Let us mention that for the classical Lorenz system λ = 0.905 and hence the coupled Lorenz system is much harder to simulate.

3. Taylor series algorithm for coupled Lorenz system

Let us denote the normalized i-th derivatives (the derivatives divided by i!) at point t with x_i, y_i, z_i, X_i, Y_i, Z_i. Then the N-th order Taylor series method with step-size $\boldsymbol{\tau}$ for system (1) is [2]:

$$x(t+\tau) \approx x_0 + \sum_{i=1}^{N} x_i\tau^i, \qquad y(t+\tau) \approx y_0 + \sum_{i=1}^{N} y_i\tau^i, \qquad z(t+\tau) \approx z_0 + \sum_{i=1}^{N} z_i\tau^i$$

$$X(t+\tau) \approx X_0 + \sum_{i=1}^{N} X_i\tau^i, \qquad Y(t+\tau) \approx Y_0 + \sum_{i=1}^{N} Y_i\tau^i, \qquad Z(t+\tau) \approx Z_0 + \sum_{i=1}^{N} Z_i\tau^i$$

The normalized derivatives are computed with the so called automatic differentiation [2]. If we apply the Leibniz rule for the derivatives of the product of two functions to system (1), we obtain the following procedure for computing them. For i = 0, … , N – 1 we compute:

$$
\begin{aligned}
&x_{i+1} = \frac{1}{i+1}a(y_i - x_i),\;\; y_{i+1} = \frac{1}{i+1}\left(r_s x_i - y_i - \sum_{j=0}^{i} x_{i-j} z_j - \varepsilon_s \sum_{j=0}^{i} X_{i-j} Y_j\right),\\
&z_{i+1} = \frac{1}{i+1}\left(\sum_{j=0}^{i} x_{i-j} y_j - b z_i\right)\\
&X_{i+1} = \frac{1}{i+1}ca(Y_i - X_i),\; Y_{i+1} = \frac{1}{i+1}\left(c\left(r_f X_i - Y_i - \sum_{j=0}^{i} X_{i-j} Z_j\right) + \varepsilon_f \sum_{j=0}^{i} X_{i-j} y_j\right),\\
&Z_{i+1} = \frac{1}{i+1}c\left(\sum_{j=0}^{i} X_{i-j} Y_j - b Z_i\right)
\end{aligned}
\tag{2}
$$

These are the formulas for the automatic differentiation. After computing the normalized derivatives (the Taylor coefficients) up to the N-th order, we evaluate the Taylor polynomials by Horner's rule. The pseudocode for the Taylor series algorithm for coupled Lorenz system is straightforward:

```
while (time < T) {
    for (i = 0; i<N; i++) {
        s1=s2=s3=s4=s5=0.0;
        for (j=0; j<=i; j++) {
            s1+= x[i-j]*z[j];   s2+= x[i-j]*y[j];
            s3+= X[i-j]*Y[j]; s4+= X[i-j]*Z[j];
            s5+= X[i-j]*y[j];
        }
        // Computing x[i+1], y[i+1], z[i+1],
        // X[i+1], Y[i+1], Z[i+1] from formulas (2)
    }
    //  Computing the optimal step-size τ
    ................................................................
    //  One step forward with Horner's rule
    //  for the new x[0], y[0], z[0], X[0], Y[0], Z[0]
    ................................................................
    time+=tau;
}
```

It is obvious that we need $O(N^2)$ floating point operations for computing the Taylor coefficients and that parallel reduction for the sums in the inner loop is the crucial part of the parallelization. To make the method more robust, we choose a simple variable step-size strategy taken from [7]:

$$
\tau = \frac{0.993}{e^2}\min\left\{\left(\frac{1}{||\mathrm{U}_{\mathrm{N}-1}||_\infty}\right)^{\frac{1}{\mathrm{N}-1}}, \left(\frac{1}{||\mathrm{U}_{\mathrm{N}}||_\infty}\right)^{\frac{1}{\mathrm{N}}}\right\}
\tag{3}
$$

where $\mathrm{U}_i = (x_i, y_i, z_i, X_i, Y_i, Z_i)$. This choice of τ ensures both the convergence of the Taylor series, and the minimization of the computational work per unit time.

4. OpenMP parallelization of the algorithm

OpenMP [8] has its own importance for the above algorithm, because: (I) OpenMP is simpler than MPI since the communication between threads is realized by the shared memory and we do not need to learn special libraries for packaging and unpackaging of multiple precision numbers. (II) OpenMP is slightly faster than pure MPI, most likely because of the additional overhead for grouping GMP data for MPI massages. (III) OpenMP uses less memory, since the algorithm does not allow domain decomposition and the computational domain has to be multiplied by the number of MPI processes. Although OpenMP has a build-in reduction clause, we cannot use it, because we use user-defined types for multiple precision numbers and user-defined operations. Thus, we have to do the reduction manually. To avoid false sharing, a padding strategy is applied for the shared containers for the partial sums of the threads [8]. The main points of the parallelization are: (I) For given **i** make an explicit parallel reduction for the sums in (2). (II) After computation of sums for given i, compute each formula for x_{i+1}, y_{i+1}, z_{i+1}, X_{i+1}, Y_{i+1}, Z_{i+1} independently in parallel. (III) After computation of all the derivatives up to the N-th, compute the variable step-size in a single section. (IV) At last use Horner's rule for evaluation of all 6 components of the solution independently in parallel. It is important to note that our approach for parallelization can be simply applied to a large class of chaotic dynamical systems, because we usually have formulas like (2), which comes from the automatic differentiation rules. The sketch of the OpenMP code for one time step in terms of GMP library reads as follows:

```
#pragma omp parallel private(i,j,tid)
{
     tid = omp_get_thread_num();
     for (i = 0; i<N; i++) {  //N - the order of the method
          # pragma omp for schedule (static)
          for (j=0; j<=i; j++) {
               mpf_mul(tempv[pad*tid],x[i-j],z[j]);
               mpf_add(sum[pad*tid],sum[pad*tid],tempv[pad*tid]);
               // The same computations for the other 4 sums
          }
         // Explicit tree based parallel Reduction
          ......................................................
          #pragma omp sections
          {
            // Computing x[i+1],y[i+1],z[i+1],X[i+1],Y[i+1],Z[i+1]
            // independently in 6 parallel sections
          }
          .......................................................
          // Setting elements of the array "sum" to zero
      }
      #pragma omp single
      {
         // Computing the variable step-size from (3)
      }
      #pragma omp sections
      {
          // One step forward with the Horner's rule
          // independently for each 6 components
      }
}
```

5. Performance and numerical results

The preparation of the parallel program and the many tests are performed in the HybriLIT platform at MLIT, JINR [9] and in the Nestum cluster, Sofia, Bulgaria [10]. Following Shijun Liao from [1], we first computed a priori estimations for the needed order N of the method and the needed decimal digits of precision K. Let Tc be the practical Lyapunov time defined by the time at which the Euclidean distance between two adjacent trajectories becomes more then 10^{-30}. We computed the following linear Tc-K and Tc-N dependencies. For fixed step-size 0.001: Tc=0.1961N-6, Tc=0.1998K-6. For variable step-size: Tc=0.1734N-6, Tc=0.1998K-6 (the same as for the fixed step-size as expected). Using these estimations we computed a reference solution in the rather long time interval [0, 400]. We took as initial conditions those from paper [11] in order to compare with the benchmark table there up to time=100. We performed two large computations, each using one CPU-node based on two Intel® Haswell processors (32 cores) in Nestum cluster: one computation with K=2158 and N=2480 and a second computation for verification with K=2254 and N=2580. We obtain numerically the following step-sizes for the second (larger) computation: min=0.001156, max=0.008854, avg=0.002855. The estimated speedup for the serial program when using variable instead of fixed step-size strategy is $2.855(0.1734/0.1961)^2$~2.23. The measured speedup is very close to that - 2.20. Because the parallel efficiency with variable step-size is slightly better (lager order of the method gives more computational work per step), the parallel program with variable step-size is 2.32 faster than the analogical fixed step-size parallel program. The needed time for the second computation using one node (32 cores) in Nestum cluster is 6.3 days. The parallel speedup when using these 32 cores is 23.1 with parallel efficiency 72.1%. As we compute the reference solution with some reserve of the estimated N and K, we actually obtain the solution with some more correct digits. The parallel program and the reference solution with 60 correct digits at every 10 time units can be seen in [12].

6.Conclusions

OpenMP parallelization of the multiple precision Taylor series method with variable step-size for the coupled Lorenz system is realized. A very good parallel efficiency for one CPU-node is observed. An important observation is that the variable step-size not only makes the method more robust and decreases the computational work, but also increases the parallel efficiency, compared to the fixed step-size case. Our approach for parallelization can be simply applied to a large class of chaotic dynamical systems. OpenMP has some advantages vice MPI, the most important of which is its simplicity, and it should be the preferred choice in the case of using a moderately large computational resource.

7. Acknowledgements

We greatly thank for the opportunity to use the computational resources of the HybriLIT Platform and Nestum cluster. The work is supported by a grant of the Plenipotentiary Representative of the Republic of Bulgaria at JINR, Dubna, Russia.

References

[1] Liao, Shijun. "On the reliability of computed chaotic solutions of non-linear differential equations." Tellus A: Dynamic Meteorology and Oceanography 61.4 (2008): 550-564.

[2] Barrio, Roberto. "Performance of the Taylor series method for ODEs/DAEs." Applied Mathematics and Computation 163.2 (2005): 525-545.

[3] Hristov, Ivan, et al. "On the efficient parallel computing of long term reliable trajectories for the Lorenz system." *arXiv preprint arXiv:2101.06682* (2021).

[4] https://gmplib.org/ (accessed 02.08.21)

[5] Dimova, Stefka, Hristov, Ivan, et al. "OpenMP parallelization of multiple precision Taylor series method." *arXiv preprint arXiv:1908.09301* (2019).

[6] Boffetta, Guido, et al. "An extension of the Lyapunov analysis for the predictability problem." Journal of the Atmospheric Sciences 55.23 (1998): 3409-3416.

[7] Jorba, Angel, and Maorong Zou. "A software package for the numerical integration of ODEs by means of high-order Taylor methods." Experimental Mathematics 14.1 (2005): 99-117.

[8] Chapman, Barbara, Gabriele Jost, and Ruud Van Der Pas. Using OpenMP: portable shared memory parallel programming. Vol. 10. MIT press, 2008.

[9] http://hlit.jinr.ru/ (accessed 02.08.21)

[10] http://hpc-lab.sofiatech.bg/ (accessed 02.08.21)

[11] Wang, Pengfei, et al. "Clean numerical simulation for some chaotic systems using the parallel multiple-precision Taylor scheme." Chinese science bulletin 59.33 (2014): 4465-4472.

[12] https://github.com/rgoranova/coupledlorenz